

# 2018 Machine Learning with R

## k-Nearest Neighbor Learning

강 필 성

고려대학교 산업경영공학부

pilsung\_kang@korea.ac.kr

# 목차

- I k-인접이웃 (분류)
- II 분류 모델 성능 평가
- III k-인접이웃 (회귀)
- IV R 실습

# 분류 문제 예시



Men

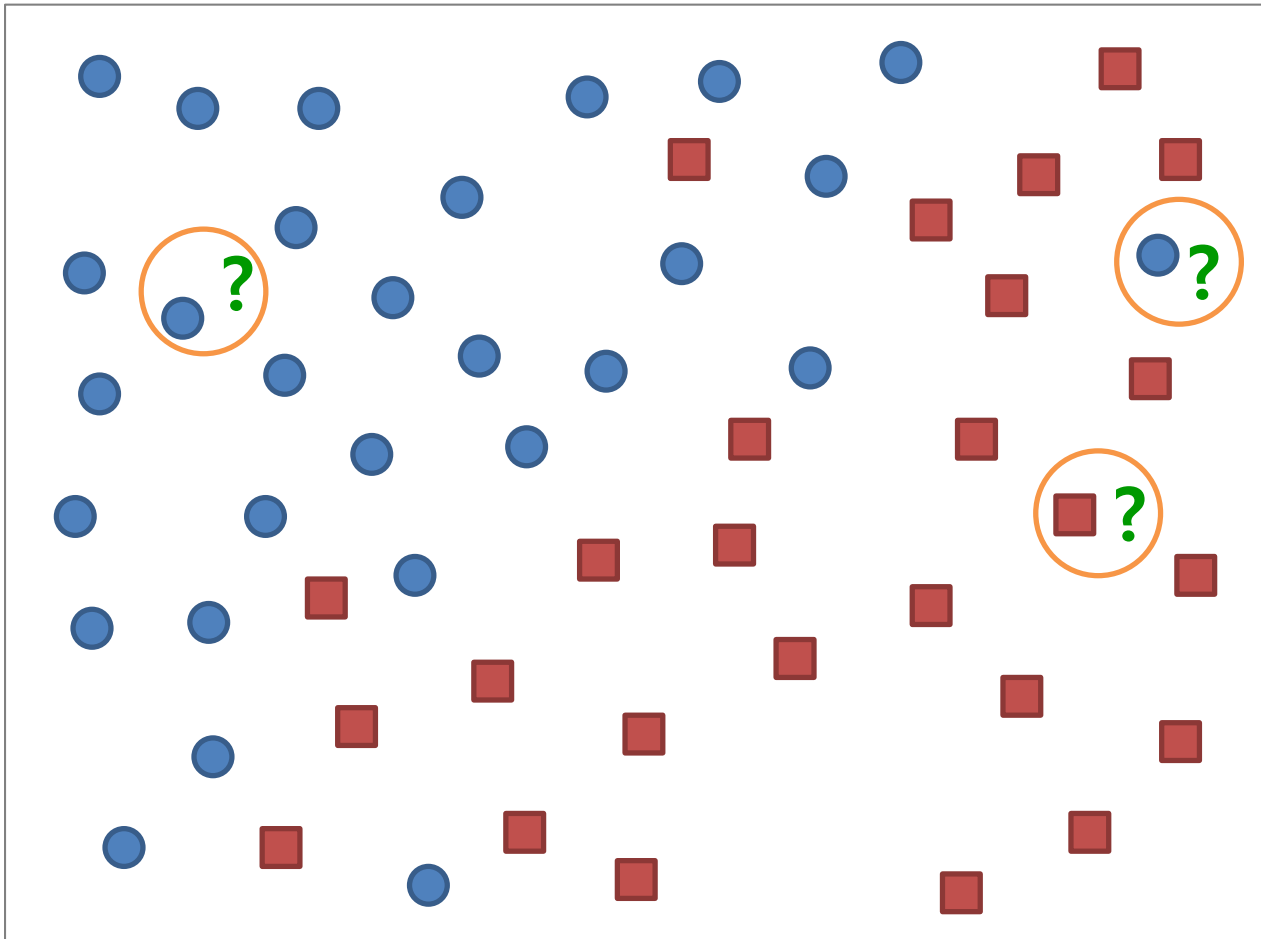
Vs.

Women



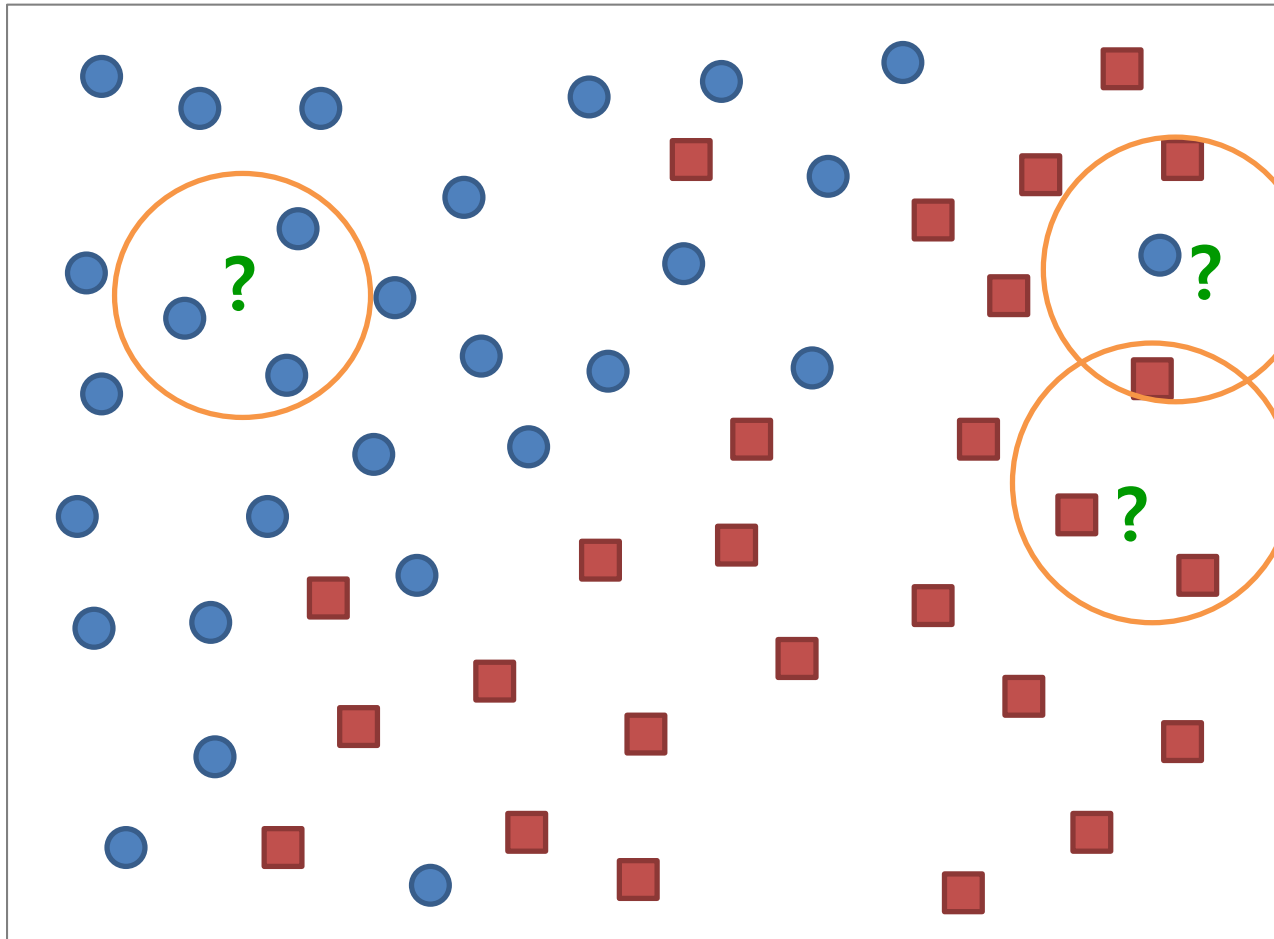
# k-인접 이웃 분류: k-Nearest Neighbor Classification

❖ 아래 물음표는 어느 범주에 속해야 하는가?



# k-인접 이웃 분류: k-Nearest Neighbor Classification

❖ 아래 물음표는 어느 범주에 속해야 하는가?



# k-인접 이웃 분류: k-Nearest Neighbor Classification

## ❖ Motivation

類類相從 近墨者黑

“Birds of a feather flock together”

“A Man is known by the company he keeps”

# k-인접 이웃 분류 절차

I

## 참조 데이터(Reference data) 준비

- 속성 정의
  - ✓ 키, 몸무게, 체지방률
- 각 범주로부터 충분한 수의 레코드 수집

개체	키	몸무게	체지방률	성별
1	187	93	15	M
2	165	51	25	F
3	174	68	14	M
4	156	48	29	F
...	...	...	...	...
N	168	59	12	M

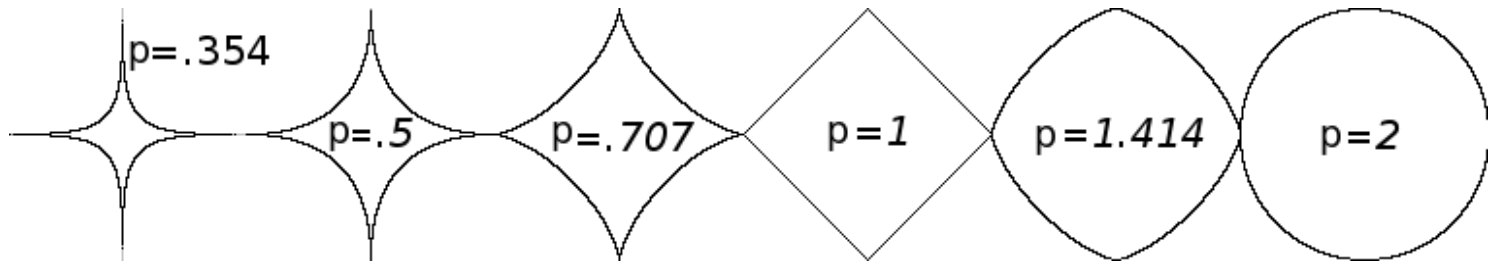
# k-인접 이웃 분류 절차

2

## 유사도 지표 정의

- 유사도는 거리에 반비례:  $\text{Similarity} \propto 1/\text{distance}$
- Minkovski distance with order  $p$

$$\text{distance}(P = (x_1, x_2, \dots, x_n), Q = (y_1, y_2, \dots, y_n)) = \left( \sum_{i=1}^n |x_i - y_i|^p \right)^{\frac{1}{p}}$$



- $p=1$ 일 때, 맨하탄 거리(Manhattan distance)
- $p=2$ 일 때, 유클리디언 거리(Euclidean distance)



# k-인접 이웃 분류 절차

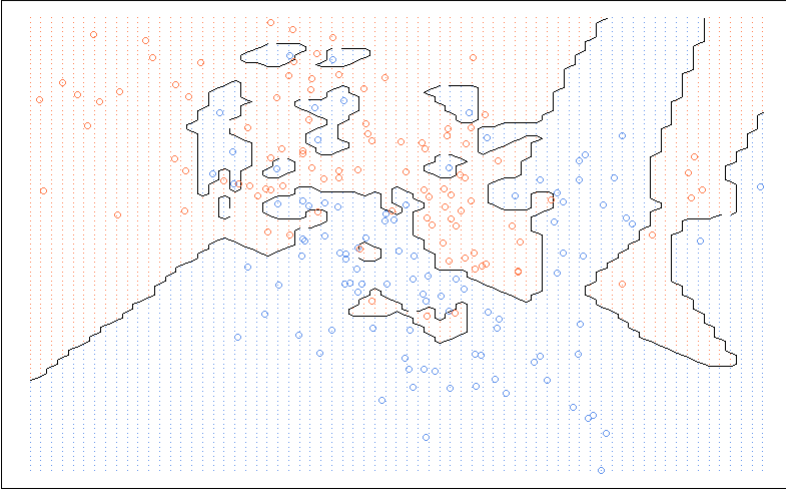
3

## k의 후보 집합 생성

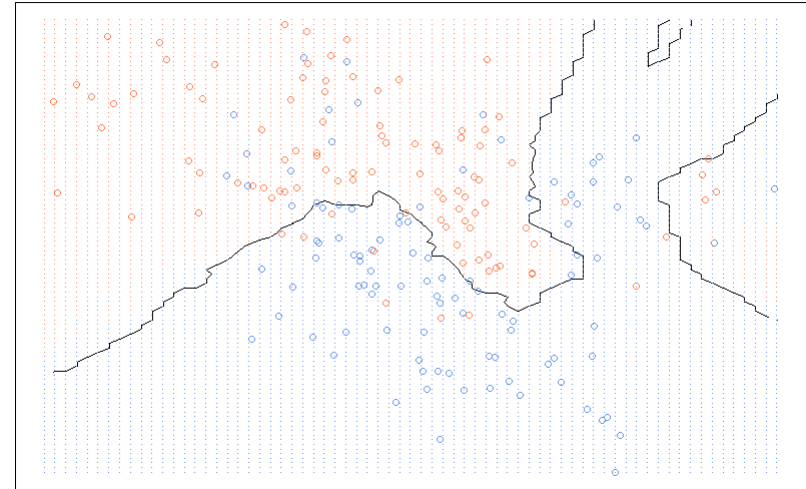
- 만일 k가 매우 작으면 노이즈에 민감한 과적합의 우려가 있음 (highly locally sensitive, over-fitting)
- 만일 k가 매우 크면 지역적 구조를 파악할 수 있는 능력을 잃게 됨(lose the ability to capture the local structure)
- 적절한 k를 찾아내는 것이 우수한 k-인접이웃 모델을 만드는 데 필수적인 요소임
- 검증 데이터에 대한 에러가 가장 낮은 k값을 선택

# k-인접 이웃 분류 절차: k에 따른 분류 경계면의 차이

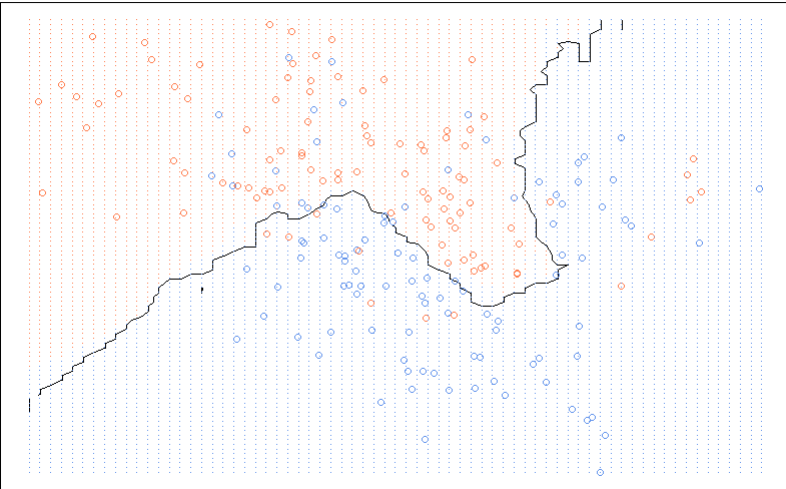
1-nearest neighbour



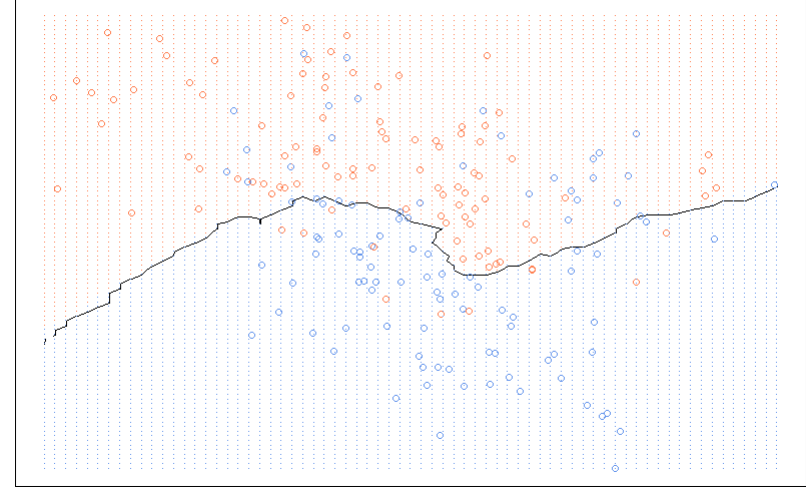
10-nearest neighbour



20-nearest neighbour

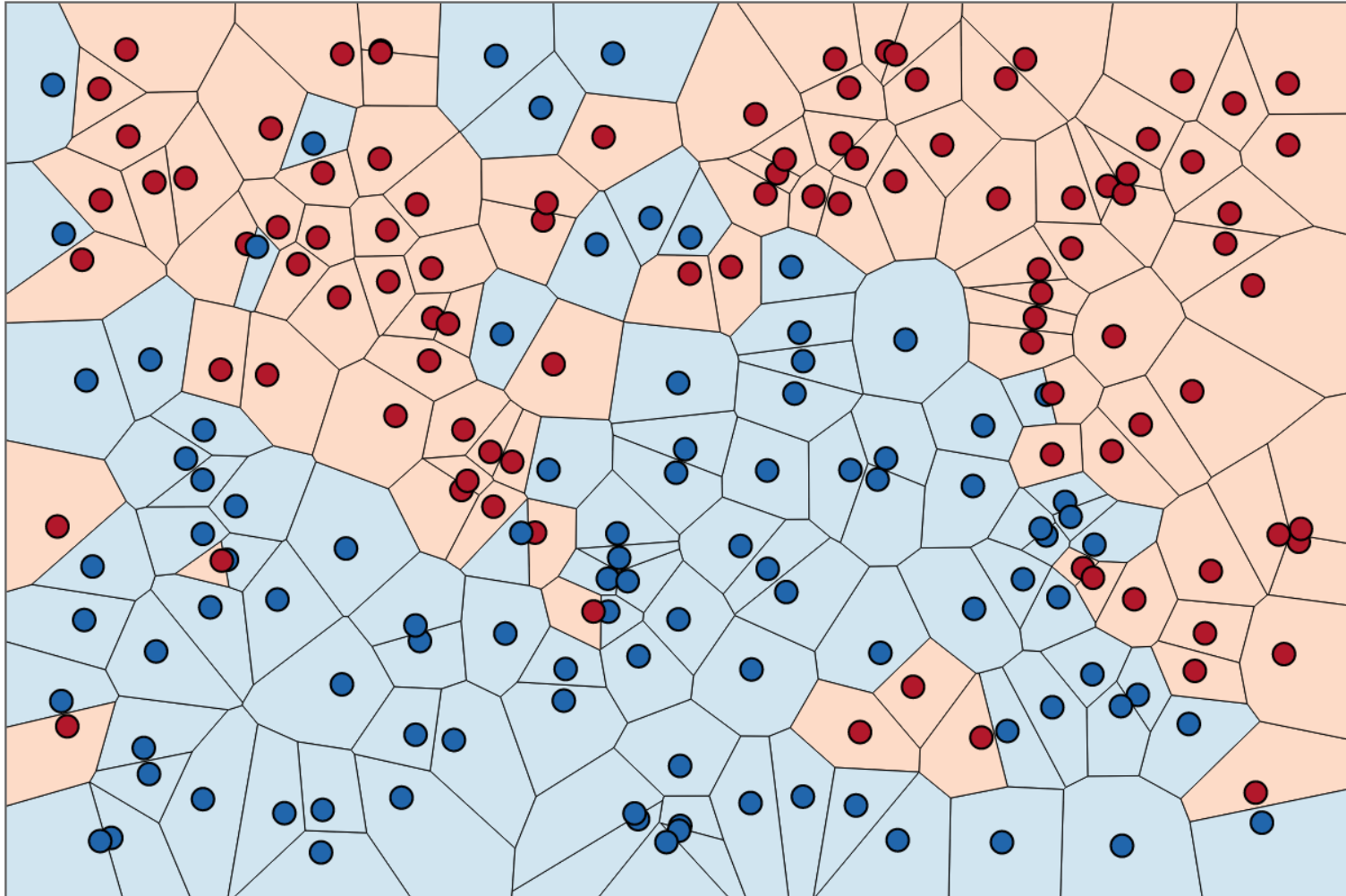


50-nearest neighbour



# k-인접 이웃 분류 절차: k에 따른 분류 경계면의 차이

❖ k=1일때 각 Reference example에 할당되는 영역



# k-인접 이웃 분류 절차

## 판별된 k개의 이웃의 범주 정보를 결합

- 다수결(Majority voting) vs. 가중합(Weighted voting)

For a  
new data

X

이웃	범주	거리	1/거리	가중치
N1	M	1	1.00	0.44
N2	F	2	0.50	0.22
N3	M	3	0.33	0.15
N4	F	4	0.25	0.11
N5	F	5	0.20	0.08

- 다수결(Majority voting):  $P(X=M) = 2/5 = 0.4$
- 가중합(Weighted voting):  $P(X=F) = 0.59$
- 분류 기준점을 0.5로 설정할 경우, 새로운 개체 X는 **다수결에 의해 여성**으로 판별되고 **가중합에 의해서는 남성**으로 판별됨

# k-인접 이웃 분류 절차

## 분류 기준값(cut-off) 설정

- 각 범주의 사전 확률(prior probability)을 고려하는 것이 바람직함
- 참조데이터셋에 각 범주의 수가  $N(C_M) = 100$ ,  $N(C_F) = 400$  일 경우,

For a new data

**X**

Neighbor	Class
N1	M
N2	F
N3	M
N4	F
N5	F

Majority voting  
 $P(X=M)=0.4$

5

- 분류 기준값이 0.5로 설정된 경우 (범주간 사전 확률이 동일), X는 여성으로 분류됨
- 분류 기준값이 0.2(남성의 사전 확률)로 설정된 경우, X는 남성으로 분류됨

# k-인접 이웃 분류 절차

검증 데이터를 이용하여 최적의 k값 결정

Value of k	% Error Training	% Error Validation
1	0.00	33.33
2	16.67	33.33
3	11.11	33.33
4	22.22	33.33
5	11.11	33.33
6	27.78	33.33
7	22.22	33.33
8	22.22	16.67
9	22.22	16.67
10	22.22	16.67
11	16.67	33.33
12	16.67	16.67
13	11.11	33.33
14	11.11	16.67
15	5.56	33.33
16	16.67	33.33
17	11.11	33.33
18	50.00	50.00

<--- Best k

# 목차

I

k-인접이웃 (분류)

II

분류 모델 성능 평가

III

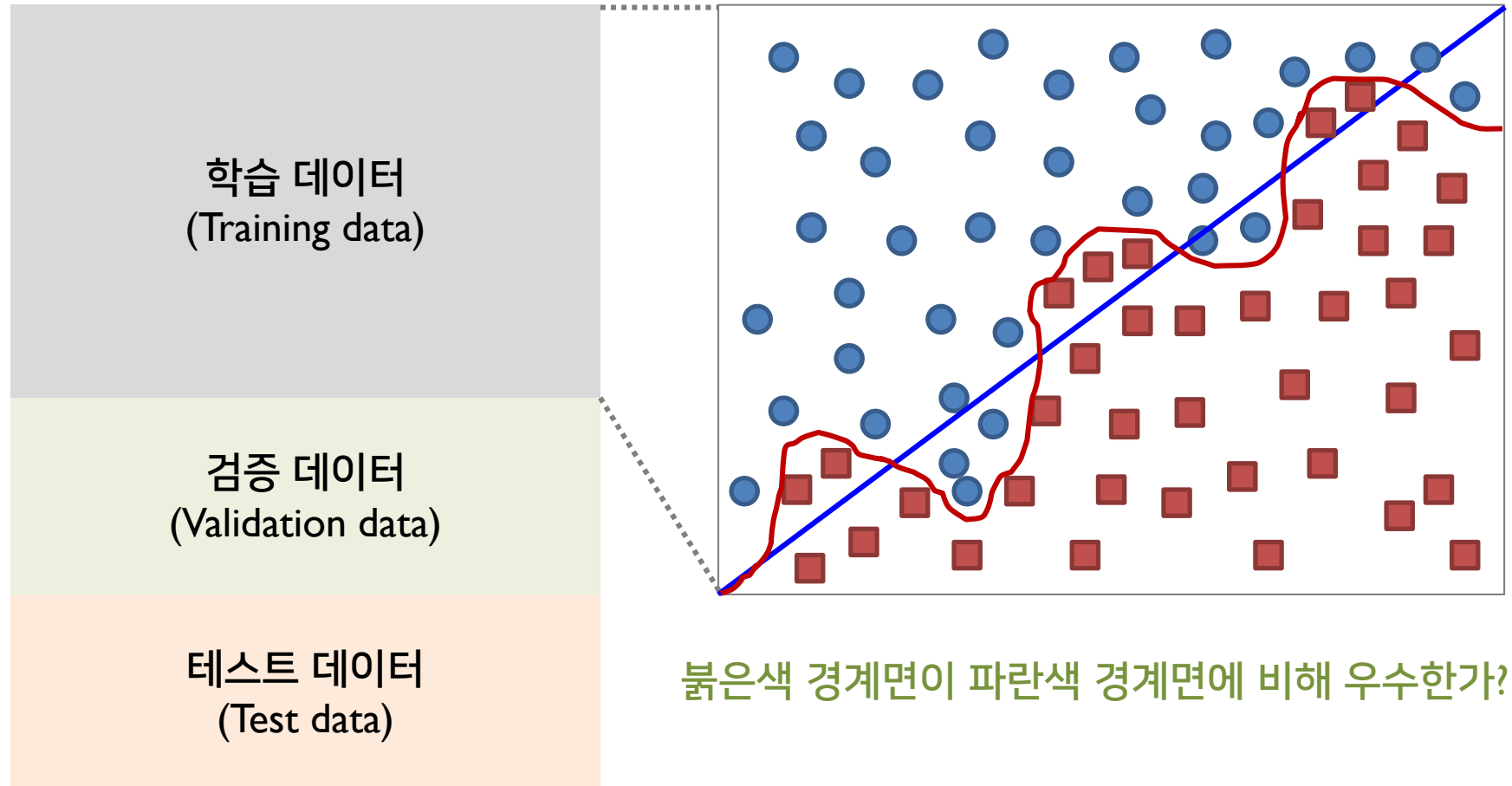
k-인접이웃 (회귀)

IV

R 실습

# 모델 평가의 필요성

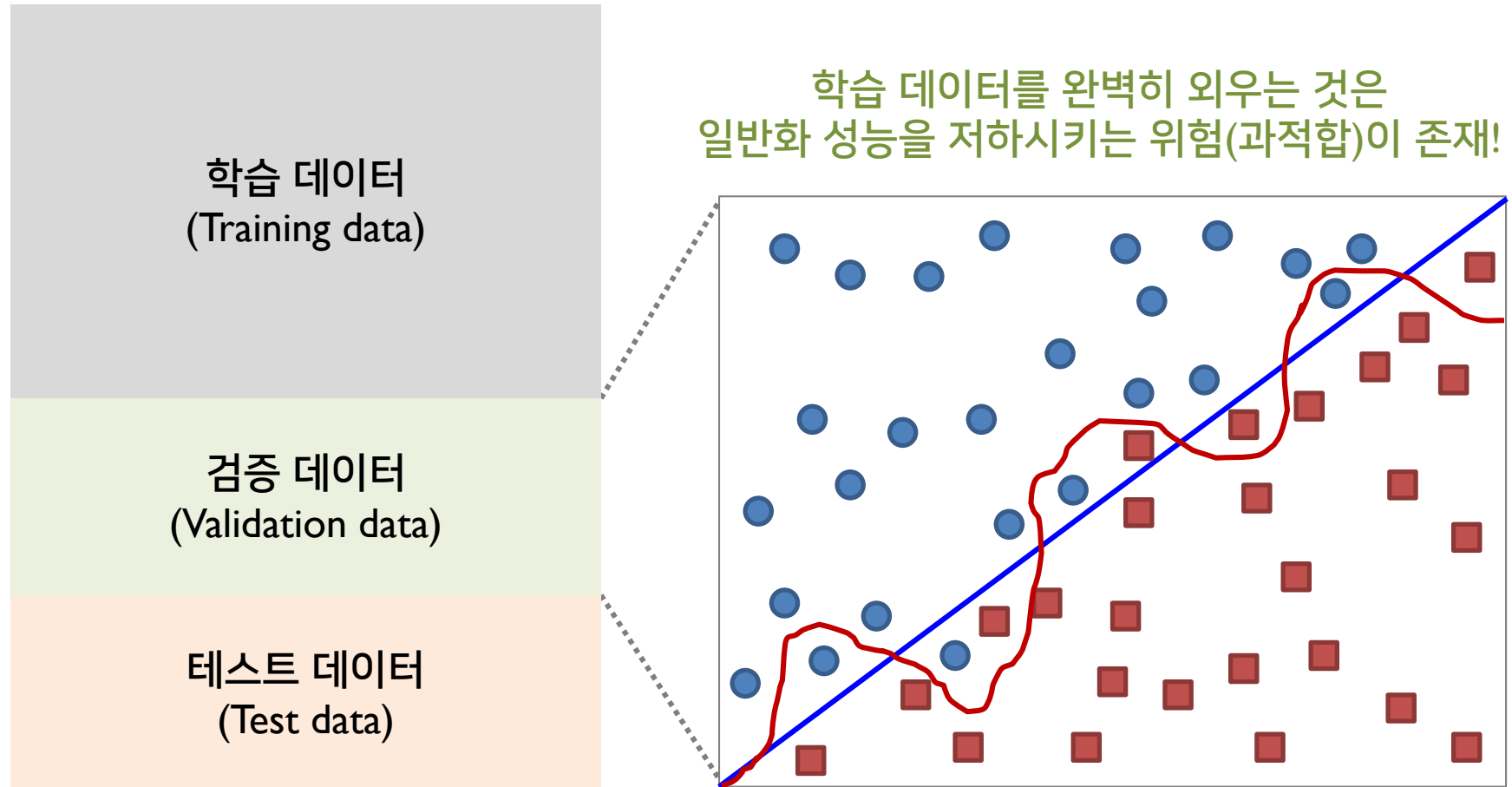
## ❖ 학습 데이터에 대한 과적합의 위험 존재





# 모델 평가의 필요성

## ❖ 학습 데이터에 대한 과적합의 위험 존재



# 모델 평가의 필요성

## ❖ 분류 문제나 회귀 문제를 풀 수 있는 다양한 알고리즘 존재

- Classification:

- ✓ Naïve bayes, linear discriminant, k-nearest neighbor, classification trees, etc.

- Prediction:

- ✓ Multiple linear regression, neural networks, regression trees, etc.

## ❖ 어떤 알고리즘은 최적의 파라미터 설정이 필요함

- k-인접이웃기법: 이웃 개체의 수(k), 인공 신경망: 은닉 노드의 수 등

## ❖ 주어진 문제를 해결하기 위한 최적의 방법론을 선택하기 위해 개별 모델을 동등한 조건에서 평가할 필요가 있음

- 검증 데이터: 다양한 파라미터 조합 중 최적의 파라미터를 찾는 데 주로 사용

- 테스트 데이터: 여러 기계학습 알고리즘 중 최적의 알고리즘을 찾는데 주로 사용

# 분류 알고리즘 평가

## 예시: 성별 분류

- 한 사람의 체지방률만을 이용하여 남성/여성 분류

									
10.0	21.7	8.9	19.9	23.4	28.9	15.7	21.6	21.5	23.2

- 단순 분류기: 체지방률이 20보다 크면 여성으로, 작으면 남성으로 분류

									
10.0	21.7	8.9	19.9	23.4	28.9	15.7	21.6	21.8	23.2
M	F	M	M	F	F	M	F	F	F


- 위 분류기의 성능을 어떻게 평가할 것인가?

# 분류 알고리즘 평가

2

## 혼동 행렬(Confusion Matrix)

- 실제 범주와 예측된 범주를 이용하여 생성한 2X2 행렬

									
10.0	21.7	8.9	19.9	23.4	28.9	15.7	21.6	21.5	23.2
M	F	M	M	F	F	M	F	F	F

- 위 결과에 대한 혼동 행렬은 다음과 같이 생성됨

Confusion Matrix		Predicted	
		F	M
Actual	F	4	1
	M	2	3

# 분류 알고리즘 평가

2

## 혼동 행렬(Confusion Matrix)

- 혼동행렬을 통해 다음과 같이 다양한 분류 성능 평가 지표를 계산할 수 있음

Confusion Matrix		Predicted	
		I (+)	0 (-)
Actual	I (+)	$n_{11}$	$n_{10}$
	0 (-)	$n_{01}$	$n_{00}$

- 민감도(Sensitivity), true positive, 재현율(recall) =  $n_{11}/(n_{11}+n_{10})$
- 특이도(Specificity, true negative) =  $n_{00}/(n_{01}+n_{00})$
- 정밀도(Precision) =  $n_{11}/(n_{11}+n_{01})$
- 제1종 오류(Type I error, false negative) =  $n_{10}/(n_{11}+n_{10})$
- 제2종 오류(Type II error, false positive) =  $n_{01}/(n_{01}+n_{00})$

# 분류 알고리즘 평가

2

## 혼동 행렬(Confusion Matrix)

Confusion Matrix		Predicted	
		1(+)	0(-)
Actual	1(+)	$n_{11}$	$n_{10}$
	0(-)	$n_{01}$	$n_{00}$

- 오분류율(Misclassification error) =  $(n_{01} + n_{10}) / (n_{11} + n_{10} + n_{01} + n_{00})$
- 정분류율(Accuracy = 1 - misclassification error) =  $(n_{11} + n_{00}) / (n_{11} + n_{10} + n_{01} + n_{00})$
- 균형 정확도 (Balanced correction rate) =  $\sqrt{\frac{n_{11}}{n_{11} + n_{10}} \cdot \frac{n_{00}}{n_{01} + n_{00}}}$
- F1 measure (정밀도와 재현율의 조화평균) =  $F1 \text{ measure} = \frac{2 \times \text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}}$

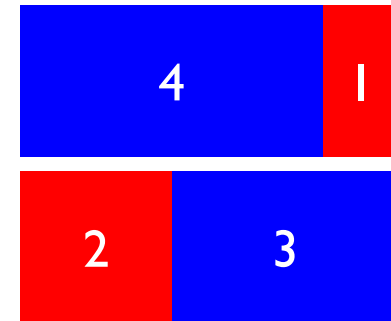
# 분류 알고리즘 평가

2

## 혼동 행렬(Confusion Matrix)

- 이전 예시에서 여성(F)을 1(+) 범주로 정의할 경우,

Confusion Matrix		Predicted	
		F	M
Actual	F	4	1
	M	2	3



- Sensitivity:  $4/5 = 0.8$ , Specificity:  $3/5 = 0.6$
- Recall:  $4/5 = 0.8$ , Precision:  $4/6 = 0.67$
- Type I error:  $1/5 = 0.2$ , Type II error:  $2/5 = 0.4$
- Misclassification error:  $(1+2)/(4+1+2+3) = 0.3$ , accuracy = 0.7
- Balanced correction rate:  $\sqrt{0.8 \times 0.6} = 0.69$
- F1 measure:  $(2 \times 0.8 \times 0.67) / (0.8 + 0.67) = 0.85$

# 분류 알고리즘 평가

3

## 분류 알고리즘의 Cut-off 설정

- 새로운 분류기: 체지방률이 0보다 크면 여성으로 분류



- 레코드들을 체지방률의 내림차순으로 정렬



- 분류를 위한 최적의 cut-off를 어떻게 설정할 것인가?



# 분류 알고리즘 평가

3

## 분류 알고리즘의 Cut-off 설정

### ▪ 다양한 Cut-off에 따른 분류 성능 비교

No.	체지방률	성별
1	28.6	F
2	25.4	M
3	24.2	F
4	23.6	F
5	22.7	F
6	21.5	M
7	19.9	F
8	15.7	M
9	10.0	M
10	8.9	M

▪ If  $\theta = 24$ ,

Confusion Matrix		Predicted	
		F	M
Actual	F	2	3
	M	1	4

- Misclassification error: 0.4
- Accuracy: 0.6
- Balanced correction rate: 0.57
- FI measure = 0.5

# 분류 알고리즘 평가

3

## 분류 알고리즘의 Cut-off 설정

### ▪ 다양한 Cut-off에 따른 분류 성능 비교

No.	체지방률	성별
1	28.6	F
2	25.4	M
3	24.2	F
4	23.6	F
5	22.7	F
6	21.5	M
7	19.9	F
8	15.7	M
9	10.0	M
10	8.9	M

▪ If  $\theta = 22$ ,

Confusion Matrix		Predicted	
		F	M
Actual	F	4	1
	M	1	4

- Misclassification error: 0.2
- Accuracy: 0.8
- Balanced correction rate: 0.8
- FI measure = 0.8

# 분류 알고리즘 평가

3

## 분류 알고리즘의 Cut-off 설정

### ▪ 다양한 Cut-off에 따른 분류 성능 비교

No.	체지방률	성별
1	28.6	F
2	25.4	M
3	24.2	F
4	23.6	F
5	22.7	F
6	21.5	M
7	19.9	F
8	15.7	M
9	10.0	M
10	8.9	M

▪ If  $\theta = 18$ ,

Confusion Matrix		Predicted	
		F	M
Actual	F	5	0
	M	2	3

- Misclassification error: 0.2
- Accuracy: 0.8
- Balanced correction rate: 0.77
- F1 measure = 0.83

# 분류 알고리즘 평가

3

## 분류 알고리즘의 Cut-off 설정

- 일반적으로 분류 알고리즘은 특정 범주에 속할 확률(probability)이나 우도(likelihood)값을 생성함
- 동일한 확률값 하에서도 Cut-off가 어떻게 설정되느냐에 따라서 분류 성능이 크게 좌우되는 상황이 발생할 수 있음
- 분류 알고리즘간의 정확한 비교를 위해서는 cut-off에 독립적인 측정 지표가 필요함
- 리프트 도표(Lift charts), receiver operating characteristic (ROC) curve 등이 사용

# Receiver Operating Characteristic (ROC) Curve

## ROC Curve 예시

- 암 진단 문제:
  - 환자의 악성 종양(malignant) 여부를 판별
  - 총 100명의 환자 중 20명의 환자가 악성 종양을 가지고 있음
  - 악성 종양 확률: 0.2

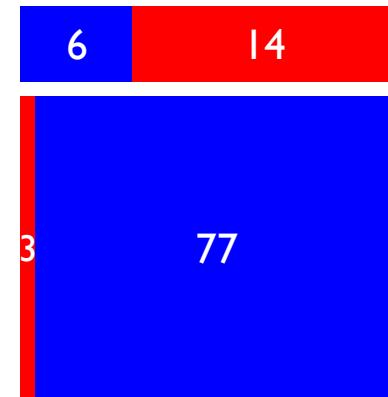
Patient	P(Malignant)	Status	Patient	P(Malignant)	Status	Patient	P(Malignant)	Status	Patient	P(Malignant)	Status
1	0.976	1	26	0.716	1	51	0.410	0	76	0.186	0
2	0.973	1	27	0.676	0	52	0.406	1	77	0.183	0
3	0.971	0	28	0.672	0	53	0.378	0	78	0.178	0
4	0.967	1	29	0.662	0	54	0.376	0	79	0.178	0
5	0.937	0	30	0.647	0	55	0.362	0	80	0.173	0
6	0.936	1	31	0.640	1	56	0.355	0	81	0.170	0
7	0.929	1	32	0.625	0	57	0.343	0	82	0.133	0
8	0.927	0	33	0.624	0	58	0.338	0	83	0.120	0
9	0.923	1	34	0.613	1	59	0.335	0	84	0.119	0
10	0.898	0	35	0.606	0	60	0.334	0	85	0.112	0
11	0.863	1	36	0.604	0	61	0.328	0	86	0.093	0
12	0.863	1	37	0.601	0	62	0.313	0	87	0.086	0
13	0.859	0	38	0.594	0	63	0.285	1	88	0.079	0
14	0.855	0	39	0.578	0	64	0.274	0	89	0.071	0
15	0.847	1	40	0.548	0	65	0.274	0	90	0.069	0
16	0.847	1	41	0.539	1	66	0.272	0	91	0.047	0
17	0.837	0	42	0.525	1	67	0.267	0	92	0.029	0
18	0.833	0	43	0.524	0	68	0.265	0	93	0.028	0
19	0.814	0	44	0.514	0	69	0.237	0	94	0.027	0
20	0.813	0	45	0.510	0	70	0.217	0	95	0.022	0
21	0.793	1	46	0.509	0	71	0.213	0	96	0.019	0
22	0.787	0	47	0.455	0	72	0.204	1	97	0.015	0
23	0.757	1	48	0.449	0	73	0.201	0	98	0.010	0
24	0.741	0	49	0.434	0	74	0.200	0	99	0.005	0
25	0.737	0	50	0.414	0	75	0.193	0	100	0.002	0

# Receiver Operating Characteristic (ROC) Curve

## 혼동행렬

- Cut-off를 0.9로 설정
  - Malignant if  $P(\text{Malignant}) > 0.9$ , else benign.

Confusion Matrix		Predicted	
		M	B
Actual	M	6	14
	B	3	77



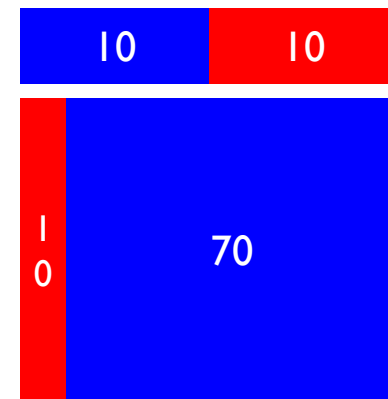
- Misclassification error = 0.17
- Accuracy = 0.83
- 이 모델은 우수한 분류 모델인가?

# Receiver Operating Characteristic (ROC) Curve

## 혼동행렬

- Cut-off를 0.8로 설정
  - Malignant if  $P(\text{Malignant}) > 0.8$ , else benign.

Confusion Matrix		Predicted	
		M	B
Actual	M	10	10
	B	10	70



- Misclassification error = 0.2
- Accuracy = 0.8
- 이 모델은 이전 모델에 비해 열등한 모델인가?



# Receiver Operating Characteristic (ROC) Curve

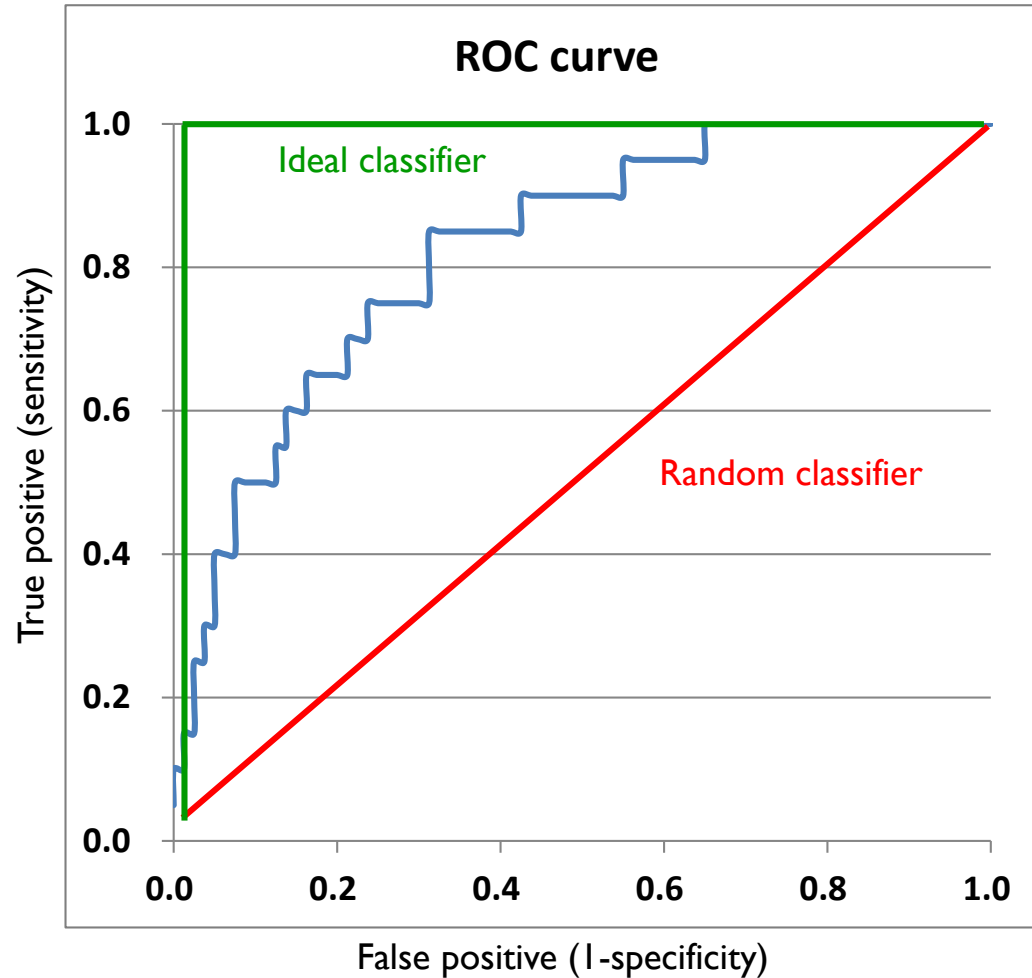
## ROC 생성 절차

- 모든 개체를 P(interesting class)를 기준으로 내림차순 정렬
- 가능한 모든 Cut-off 경우에 대해 True Positive Rate와 False Positive Rate를 계산
- X축이 False Positive Rate, Y축이 True Positive Rate가 되는 2차원 그래프 도시

Patient	P(Malignant)	Status	True positive	false positive
1	0.976	1	0.050	0.000
2	0.973	1	0.100	0.000
3	0.971	0	0.100	0.013
4	0.967	1	0.150	0.013
5	0.937	0	0.150	0.025
6	0.936	1	0.200	0.025
7	0.929	1	0.250	0.025
8	0.927	0	0.250	0.038
⋮	⋮	⋮	⋮	⋮
96	0.019	0	1.000	0.950
97	0.015	0	1.000	0.963
98	0.010	0	1.000	0.975
99	0.005	0	1.000	0.988
100	0.002	0	1.000	1.000

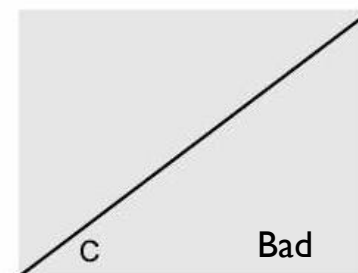
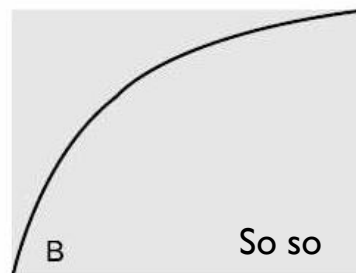
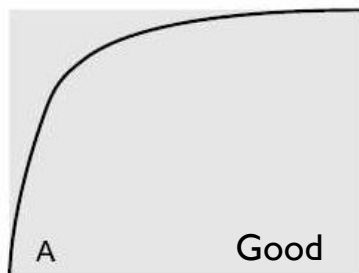
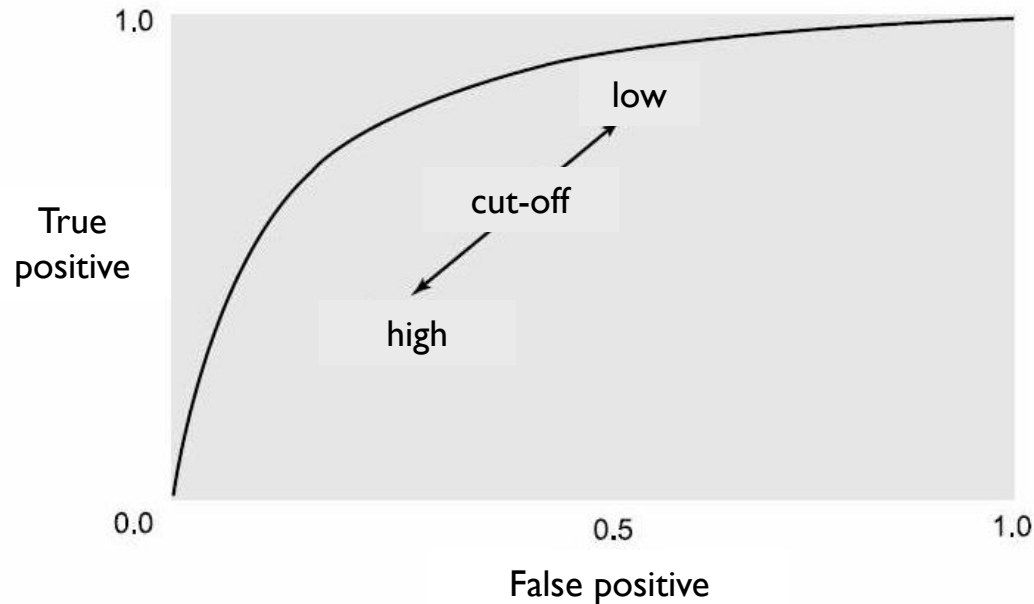
# Receiver Operating Characteristic (ROC) Curve

## ROC Curve 범위



# Receiver Operating Characteristic (ROC) Curve

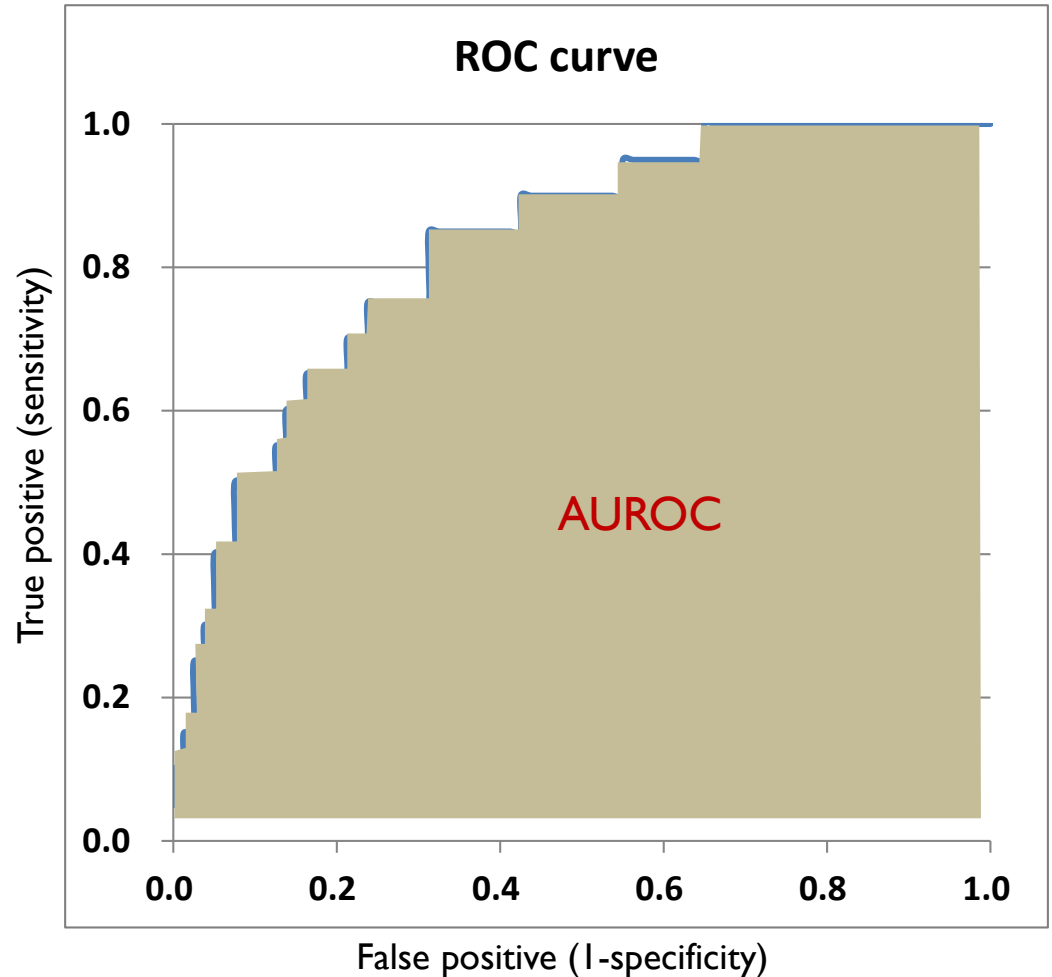
## ROC Curve와 cut-off와의 관계



# Area Under ROC Curve: AUROC

## Area Under ROC curve (AUROC)

- ROC curve 아래의 면적
- 이상적인 분류기는 1의 값을 갖고, 무작위 분류기는 0.5의 값을 가짐
- Cut-off에 독립적인 알고리즘 성능 평가 지표로 사용될 수 있음



# 비대칭 오분류 비용

## 비대칭 오분류 비용: Asymmetric misclassification costs

- 많은 비즈니스 문제에서는 한 범주를 정확하게 판별하는 것이 다른 범주를 정확하게 판별하는 것보다 중요하게 취급됨
  - 질병 진단, 세금 사기, 신용카드 사기, 마케팅 프로모션 응답 예측 등
- 이러한 경우에는 전체 집합에 대한 오분류가 증가하더라도 주요 범주에 대한 정확도를 높이는 것이 효과적일 수 있음
  - 특정 범주에 대한 오분류 비용이 다른 범주보다 큰 경우
  - 특정 범주에 대한 정분류 효과가 다른 범주보다 큰 경우

# 비대칭 오분류 비용

## 예시: 마케팅 반응 모델링

- 1000명의 고객에게 마케팅 프로모션을 진행함 (반응률은 1%, “1” = response, “0” = non-response).
- 단순 규칙: Naïve rule
  - 모든 고객이 반응하지 않을 것(0)으로 예측하자!.

Confusion Matrix		Predicted	
		1	0
Actual	1	0	10
	0	0	990

- Misclassification error = 1%
- Accuracy = 99%.

# 비대칭 오분류 비용

## 예시: 마케팅 반응 모델링

### ▪ 기계학습 모델

- 실제 반응 고객 10명 중 8명을 정확하게 예측했으나, 실제로 반응하지 않은 20명의 고객을 반응할 것으로 예측함

Confusion Matrix		Predicted	
		1	0
Actual	1	8	2
	0	20	970

- Misclassification error = 2.2%
- Accuracy = 97.8%

- 과연 이 모델이 이전 모델보다 열등하다고 할 수 있는가?

# 비대칭 오분류 비용

## 예시: 마케팅 반응 모델링

### 정분류/오분류에 대한 편익/비용을 설정

#### 예시:

- ✓ **\$10**: 반응 고객들에게 제품을 판매하여 얻는 이득
- ✓ **-\$10**: 반응 고객을 판별하지 못해 생기는 기회비용
- ✓ **-\$1**: 반응할 것으로 예측된 고객에게 소요되는 마케팅 비용

Confusion Matrix		Predicted	
		1	0
Actual	1	<b>\$9</b>	<b>-\$10</b>
	0	<b>-\$1</b>	0

- Naïve rule 적용 시 최종 이득/비용:  $10 * (-\$10) = \textbf{-\$100}$
- 기계학습 모델 적용 시 최종 이득/비용:  $8 * (\$9) + 2 * (-\$10) + 20 * (-\$1) = \textbf{\$32}$



# 비대칭 오분류 비용

## 암 진단 예측 모델의 편익/비용 행렬

- 암에 걸린 사람을 정상으로 판별한 오분류에 대한 비용을 수치화할 수 있는가?

Confusion Matrix		Predicted	
		1	0
Actual	1	Save one's life	Can measure?
	0	Misdiagnosis cost	0

- 일반적으로 의료진들이 매우 보수적인 진단을 하는 이유

# k-인접이웃 예시

## ❖ 스팸 필터링

받은 편지함 - ps kang@snu.ac.kr - Microsoft Outlook

보기

전체 회신, 전달, 모임, 이동 위치?, 관리자에게 전달, 팀 전자 메일, 완료, 회신하고 삭제, 새로 만들기, 이동, 규칙, OneNote, 읽지 않음/읽음, 범주, 추가 작업, 연락처 찾기, 주소록, 전자 메일 필터링, 모든 폴더 보내기/받기, 보내기/받기

받은 편지함 검색(Ctrl+E)

정렬 기준: 날짜 | 새로운 항목 우선

▲ 어제

정육진 (일) 오후 8:45  
RE: RE: 캡스톤디자인 4/15 미팅시간에 대한 메일입니다.

정육진 (일) 오후 8:43  
RE: RE: 캡스톤디자인 4/15 미팅시간에 대한 메일입니다.

▲ 지난 주

정육진 03-31 (목)  
캡스톤디자인 4/15 미팅시간에 대한 메일입니다.

안재경 03-31 (목)  
Fwd: Agreement between Northumbria and Seoul Tech

정준균 03-31 (목)  
안녕하세요 교수님. 데이터분석도구강의들은 07학번 정준균입니다.

송미자 03-31 (목)  
캡스톤 지도교수 명단

송미자 03-31 (목)  
캡스톤디자인 지도교수 최종 수정본 보내드립니다

Seoung Bum Kim 03-31 (목)  
INFORMS Data Mining Section Invitation

김광한 03-31 (목)  
답변 주셔서 감사합니다.

좌성훈 03-31 (목)  
교수 산악회 4월 산행 공지드립니다(4월 2일 토요일, 정계산)

INFORMS Data Mining Section Invitation

Seoung Bum Kim <sbkim1@korea.ac.kr>

2011-04-01 오전 10:14에 이 메시지에 회신했습니다.

보낸 날짜: 2011-03-31 (목) 오후 3:36

받는 사람: Pilsung Kang

강필성 교수님,

금년 INFORMS학회가 North Carolina Charlotte에서 11월 13일부터 16일까지 열립니다. <http://meetings2.informs.org/charlotte2011/>

먼저 다음주 금요일까지 이번 INFORMS에 가실 의향이 있으신지에 관해 알려주실 수 있는지요? 제 계획은 Novelty score와 관련된 talk들을 모아 한 세션을 구성하려고 합니다.

감사합니다.

김성범 드림

Seoung Bum Kim, Associate Professor  
Data Mining & Quality Management Lab.  
School of Industrial Management Engineering  
Korea University  
Anam-Dong, Seongbuk-Gu, Seoul 136-713  
<http://dmqm.korea.ac.kr/>  
02-3290-3397

# k-인접이웃 예시

## ❖ 스팸 필터링

메일

스팸 메일함 86 / 86 안읽은 메일 삭제

[안내] 중요 메일 '검토 후 발송' 기능 메일검색

중요 메일은 검토하고 보낼 수 있어요.

전체선택 삭제 스팸신고 스팸해제 전달 이동 읽음

전체선택	★	보낸사람	제목	날짜	크기
<input type="checkbox"/>	★	진재환3入	솔직한 사람들의 모임터예요	11-03-30 14:29	1.1KB
<input type="checkbox"/>	★	aezpr	한'동'안 예전처럼 지냈어요? 전혀 편지가 어려워 불안했어요	11-03-30 13:11	1.5KB
<input type="checkbox"/>	★	모든-edu	[5월12일개강확정] 평생교육사,사복2급,보육2급 100%온라인	11-03-30 09:46	3.1KB
<input type="checkbox"/>	★	박수진	실제로 안 만나셔도 되요 ~ 여자회원을 사진보는 재미로 그냥 놀러오세요	11-03-30 07:20	971B
<input type="checkbox"/>	★	jiuzju@peew.kr	프로토/토토 일확천금을 위한 베티!!!!	11-03-30 06:32	1.1KB
<input type="checkbox"/>	★	tapvmo@jocjg.co.kr	매너게임 합니다 사설 토토,프로토 vip회원 모집	11-03-30 05:55	1.1KB
<input type="checkbox"/>	★	래금	&#9670;&#49888;&#8224;&#50556;&#8224;&#47560;&#8224;&#53664;&#9670;&#33;&#65339;&#50724;&#8224;&#54536;&#8224;&#54665;&#8224;&#49324;&#8224;&#49345;&#8224;&#54408;&#8224;&#44428;&#65341;&#43;&#49;&#48;&#47560;&#45324;&#33;&#45796;&#46308;&#48512;&#51088;&#46104;&#49464;&#50857;&#126;&#9829;&#13;&#10	11-03-30 04:05	1.3KB
<input type="checkbox"/>	★	나이트팔라스	라이브카지노! 입금시 10% 서비스머니, 조작이 절대 없는 실 제 달러의 생방송 게임진행	11-03-30 03:33	3.7KB

1 2 3

# k-인접이웃 예시

## ❖ 스팸 필터링

### ■ 데이터 구성

✓ 레코드: 1개의 메일

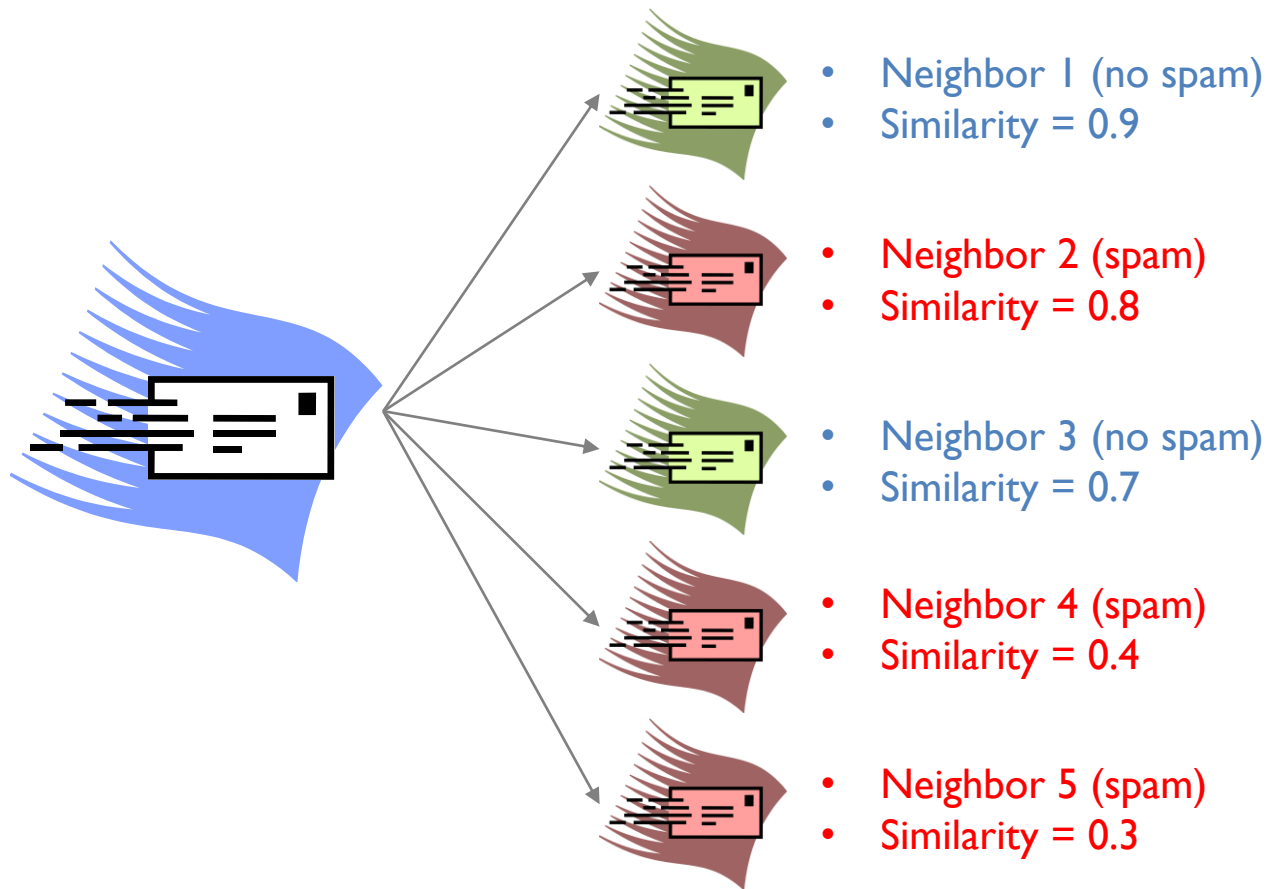
✓ 속성: 메일 본문에 나타난 키워드들의 출현 빈도

Mail	회의	수정	기안	보고	대박	머니	외로워	미팅	...	스팸?
1	2	3	1	0	0	0	0	0	...	N
2	1	0	2	3	0	0	1	0	...	N
3	2	2	3	1	0	0	0	1	...	N
4	0	0	0	0	3	2	0	0	...	Y
5	0	0	0	1	0	0	2	3	...	Y
...	...	...	...	...	...	...	...	...	...	...

# k-인접이웃 예시

## ❖ 스팸 필터링

- 스팸 필터링을 위해 새로 도착한 메일을 대상으로 5개의 유사한 메일을 비교



If we use the  
majority voting, then  
classify the mail as  
spam

If we use the  
weighted voting,  
then classify the mail  
as spam

# 목차

- I k-인접이웃 (분류)
- II 분류 모델 성능 평가
- III k-인접이웃 (회귀)
- IV R 실습

# 회귀분석



Predict  
one's BFS



# k-인접 이웃 회귀 절차

I

## 참조 데이터(Reference data) 준비

- 속성 정의
  - ✓ 키, 몸무게, 체지방률
- 각 범주로부터 충분한 수의 레코드 수집

개체	키	몸무게	성별(F=1)	체지방률
1	187	93	0	15
2	165	51	1	25
3	174	68	0	14
4	156	48	1	29
...	...	...	...	...
N	168	59	0	12



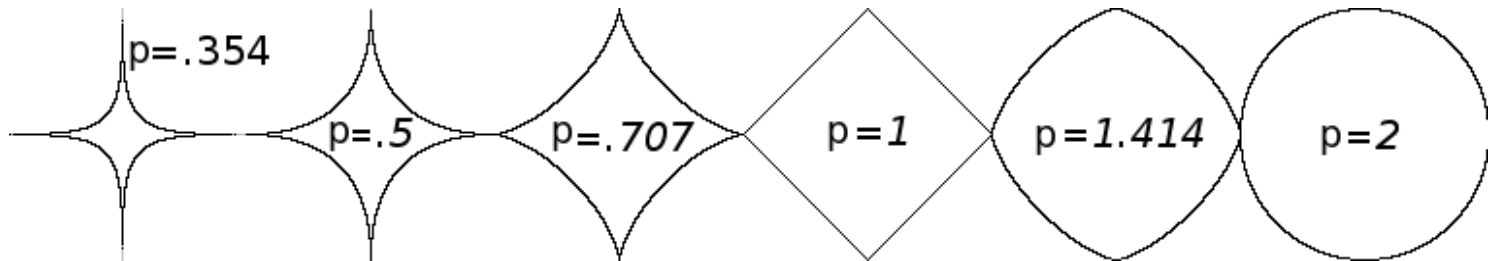
# k-인접 이웃 회귀 절차

2

## 유사도 지표 정의

- 유사도는 거리에 반비례:  $\text{Similarity} \propto 1/\text{distance}$
- Minkovski distance with order  $p$

$$\text{distance}(P = (x_1, x_2, \dots, x_n), Q = (y_1, y_2, \dots, y_n)) = \left( \sum_{i=1}^n |x_i - y_i|^p \right)^{\frac{1}{p}}$$



- $p=1$ 일 때, 맨하탄 거리(Manhattan distance)
- $p=2$ 일 때, **유클리디언 거리(Euclidean distance)**

# k-인접 이웃 회귀 절차

3

## k의 후보 집합 생성

- 만일 k가 매우 작으면 노이즈에 민감한 과적합의 우려가 있음 (highly locally sensitive, over-fitting)
- 만일 k가 매우 크면 지역적 구조를 파악할 수 있는 능력을 잃게 됨(lose the ability to capture the local structure)
- 적절한 k를 찾아내는 것이 우수한 k-인접이웃 모델을 만드는 데 필수적인 요소임
- 검증 데이터에 대한 에러가 가장 낮은 k값을 선택

# k-인접 이웃 회귀 절차

## 결합 규칙(combining rule) 정의

- 단순 평균(Simple average) vs. 가중 평균(Weighted average)

For a  
new data

X

이웃	체지방률	거리	1/거리	가중치
N1	15.4	1	1.00	0.44
N2	17.2	2	0.50	0.22
N3	12.3	3	0.33	0.15
N4	11.5	4	0.25	0.11
N5	10.9	5	0.20	0.08

- 단순 평균 이용

✓ BFS of X =  $(15.4+17.2+12.3+11.5+10.9)/5 = 13.46$

- 가중 평균 이용

✓ BFS of X =

$0.44*15.4+0.22*17.2+0.15*12.3+0.11*11.5+0.08*10.9 = 14.54$

# k-인접 이웃 회귀 절차

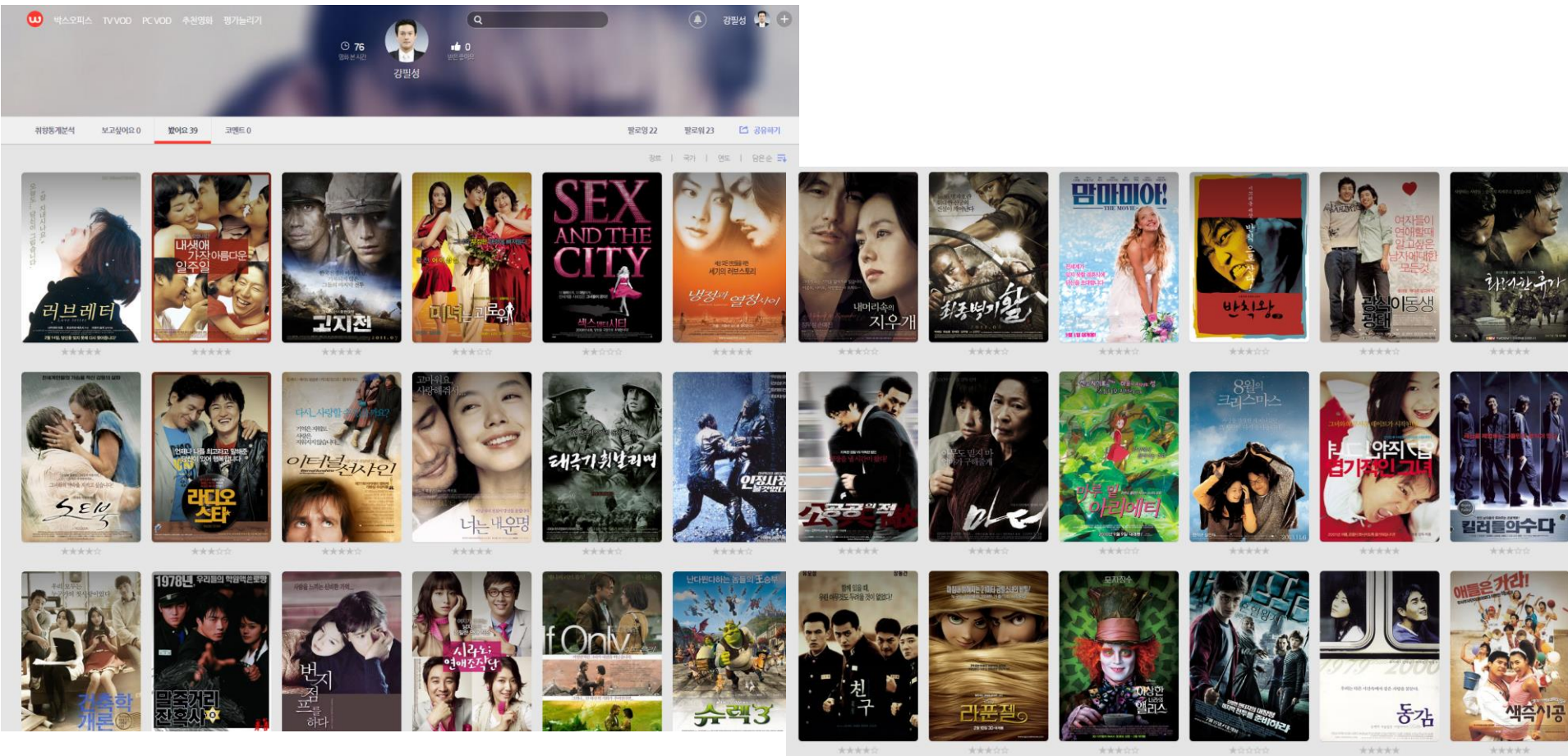
검증 데이터를 이용하여 최적의 k값 설정

Value of k	% Error Training	% Error Validation
1	0.00	33.33
2	16.67	33.33
3	11.11	33.33
4	22.22	33.33
5	11.11	33.33
6	27.78	33.33
7	22.22	33.33
8	22.22	16.67
9	22.22	16.67
10	22.22	16.67
11	16.67	33.33
12	16.67	16.67
13	11.11	33.33
14	11.11	16.67
15	5.56	33.33
16	16.67	33.33
17	11.11	33.33
18	50.00	50.00

<--- Best k

# k-인접 이웃 회귀 응용사례

## ❖ 영화 추천 시스템: 협업 필터링



<http://watcha.net>



# k-인접 이웃 회귀 응용사례

## ❖ 추천 영화 리스트

예상 별점이 가장 높은 영화 × 장르 | 국가 | 연도 | 추천 이유

<p>4.3</p> <p>예상 별점 4.3개 ★★★★★</p> <p>《내 생애 가장 아름다운 일주일》과 비슷해요.</p>	<p>4.2</p> <p>《괴물의 크리스마스》와 비슷해요.</p>	<p>4.0</p> <p>《시리노 연애조각단》과 비슷해요.</p>	<p>4.3</p> <p>《즐거이는 배우 설경구》</p>
<p>4.2</p> <p>《친구》와 비슷해요.</p>	<p>4.0</p> <p>《나는 내 운명》과 비슷해요.</p>	<p>4.0</p> <p>예상 별점 4.0개 ★★★★★</p>	<p>4.3</p> <p>예상 별점 4.3개 ★★★★★</p>
<p>4.6</p> <p>《이터널 선샤인》의 2편과 비슷해요.</p>	<p>4.0</p> <p>《우리들의 행복한 시간》과 비슷해요.</p>	<p>4.0</p> <p>《노트북》의 1편과 비슷해요.</p>	<p>4.0</p> <p>예상 별점 4.0개 ★★★★★</p>

# k-인접 이웃 회귀 응용사례

## ❖ 영화 추천 시스템: 협업 필터링

	영화 1	영화 2	영화 3	영화 4	영화 5	...	영화 D
강필성	10	9	5	6	9	...	? <b>9</b>

고객	영화 1	영화 2	영화 3	영화 4	영화 5	...	영화 D
<b>1</b>	10	8	4	7	10	...	10
2	8	5	7	9	4	...	5
<b>3</b>	10	9	6	5	8	...	9
4	4	2	10	10	5	...	3
5	7	4	6	8	5	...	3
6	5	2	10	10	10	...	6
<b>7</b>	10	8	6	6	8	...	8
...	...	...	...	...	...	...	...
N	5	7	1	5	4	...	7

# 목차

- I k-인접이웃 (분류)
- II 분류 모델 성능 평가
- III k-인접이웃 (회귀)
- IV R 실습



# R 실습: k-NN Illustration

## ❖ 필요 패키지 설치

```
# k-Nearest Neighbor Illustration -----  
install.packages("ElemStatLearn", dependencies = TRUE)  
install.packages("class", dependencies = TRUE)  
  
library(ElemStatLearn)  
library(class)
```

- “ElemStatLearn”: 인공 데이터를 이용한 k-NN 시각화 예시용
- “class”: k-NN 분류 알고리즘 제공

# R 실습: k-NN Illustration

## ❖ 필요 패키지 설치

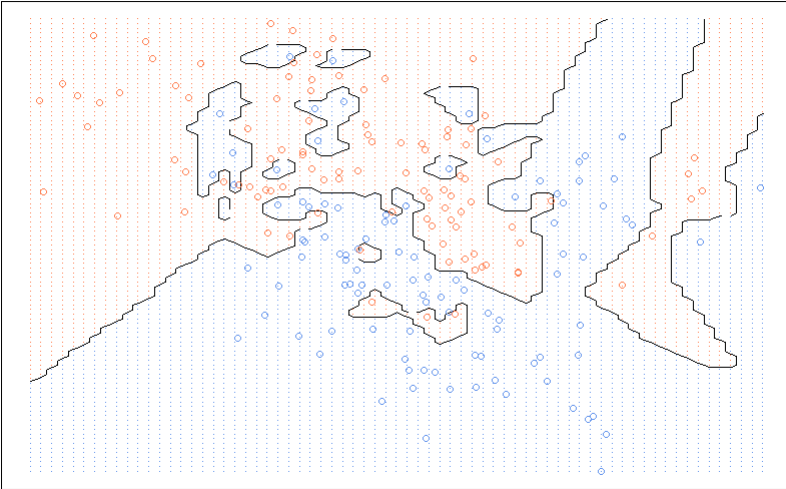
```
# 2-D artificial data example with k=1
x <- mixture.example$x
g <- mixture.example$y
xnew <- mixture.example$xnew
mod1 <- knn(x, xnew, g, k=50, prob=TRUE)
prob1 <- attr(mod1, "prob")
prob1 <- ifelse(mod1=="1", prob1, 1-prob1)
px1 <- mixture.example$px1
px2 <- mixture.example$px2
prob1 <- matrix(prob1, length(px1), length(px2))
par(mar=rep(2,4))
contour(px1, px2, prob1, levels=0.5, labels="", xlab="", ylab="", main= "50-
nearest neighbour", axes=FALSE)
points(x, col=ifelse(g==1, "coral", "cornflowerblue"))
gd <- expand.grid(x=px1, y=px2)
points(gd, pch=".", cex=1.2, col=ifelse(prob1>0.5, "coral", "cornflowerblue"))
box()
```

- Red box 안의 k값을 변경해가면서 실행

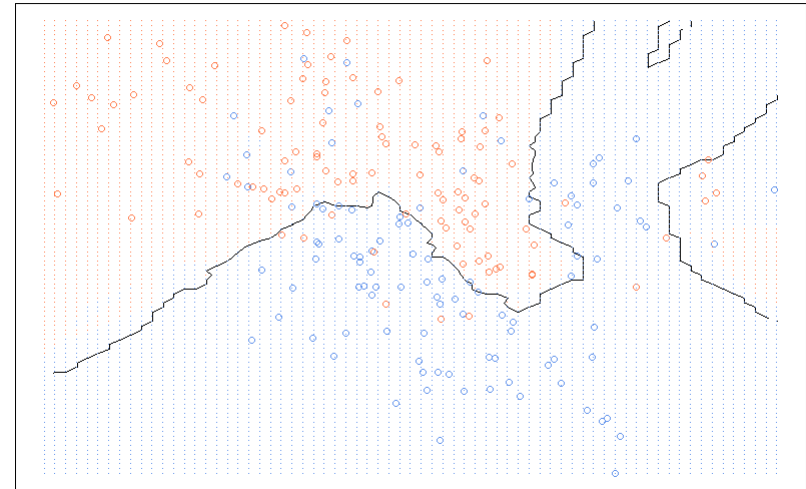
# R 실습: k-NN Illustration

## ❖ 여러 k값에 따른 분류 경계면

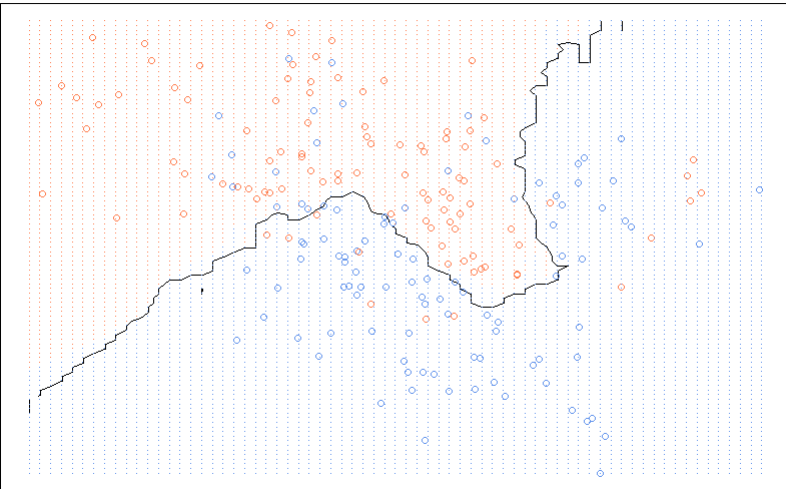
1-nearest neighbour



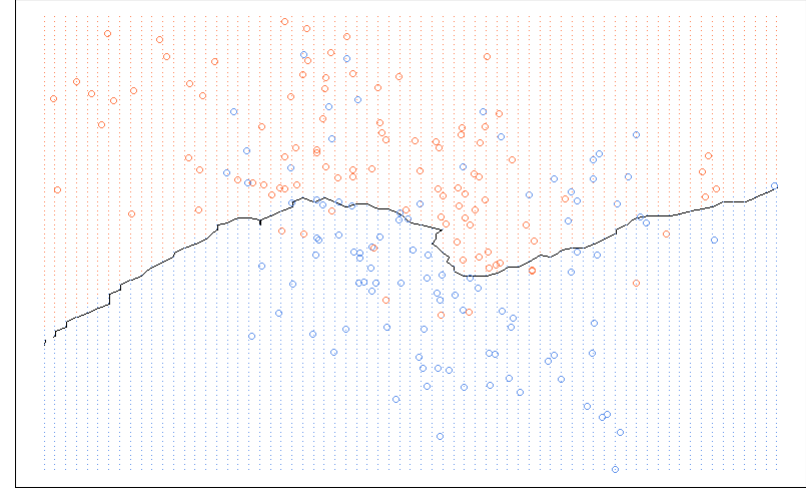
10-nearest neighbour



20-nearest neighbour



50-nearest neighbour



# R 실습: k-NN Classification

## ❖ 데이터: Wisconsin Breast Cancer 데이터

- 총 569명의 환자 (357명 악성, 212명 양성)
- 악성 종양 여부 판별
- 세포핵의 3차원 이미지에 대해 각각 다음과 같이 10가지의 값을 산출
  - ✓ a) radius (mean of distances from center to points on the perimeter)
  - ✓ b) texture (standard deviation of gray-scale values)
  - ✓ c) perimeter
  - ✓ d) area
  - ✓ e) smoothness (local variation in radius lengths)
  - ✓ f) compactness ( $\text{perimeter}^2 / \text{area} - 1.0$ )
  - ✓ g) concavity (severity of concave portions of the contour)
  - ✓ h) concave points (number of concave portions of the contour)
  - ✓ i) symmetry
  - ✓ j) fractal dimension ("coastline approximation" - 1)

# R 실습: k-NN Classification

## ❖ 필요 패키지 설치

```
# k-Nearest Neighbor Learning (Classification) -----  
# kknn package install & call  
install.packages("kknn", dependencies = TRUE)  
library(kknn)
```

- “kknn”: k-NN 분류 알고리즘과 함께 parameter search를 할 수 있는 옵션 제공

# R 실습: k-NN Classification

## ❖ 데이터 불러오기 및 전처리

```
# Load the wdbc data
RawData <- read.csv("wdbc.csv", header = FALSE)
head(RawData)

# k-NN Classification: WDBC data
# Normlaize the input data
Class <- RawData[,31]
InputData <- RawData[,1:30]
ScaledInputData <- scale(InputData, center = TRUE, scale = TRUE)
```

- head( ): 입력인자의 초기 일부분만 콘솔창에 출력
- scale( ): 정규화를 수행하는 함수
  - ✓ center = TRUE: 각 변수의 평균값이 0이 되도록 변환
  - ✓ scale = TRUE: 각 변수의 표준편차가 1이 되도록 변환
- 두 개체 사이의 거리를 구하는 절차가 포함된 Machine Learning 알고리즘은 반드시 정규화를 수행해주어야 함

# R 실습: k-NN Classification

## ❖ 데이터 불러오기 및 전처리

```
# Divide the dataset into the training (70%) and Test (30%) datasets
set.seed(12345)
trn_idx <- sample(1:length(Class), round(0.7*length(Class)))
trnInputs <- ScaledInputData[trn_idx,]
trnTargets <- Class[trn_idx]
tstInputs <- ScaledInputData[-trn_idx,]
tstTargets <- Class[-trn_idx]

trnData <- data.frame(trnInputs, trnTargets)
colnames(trnData)[31] <- "Target"
tstData <- data.frame(tstInputs, tstTargets)
colnames(tstData)[31] <- "Target"
```

- 데이터를 학습용(70%)과 테스트용(30%)로 분할

# R 실습: k-NN Classification

## ❖ k-NN 분류 수행

```
# Perform k-nn classification with  
# k=1, Distance = Euclidean, and weighted scheme = majority voting  
kknn <- kknn(Target ~ ., trnData, tstData, k=1, distance=2, kernel = "rectangular")
```

### ■ kknn( ): k-NN 수행 함수

- ✓ 첫 번째 인자: Formula
- ✓ 두 번째 인자: 학습 데이터, 세 번째 인자: 테스트 데이터
- ✓ 네 번째 인자: k
- ✓ 다섯 번째 인자: 거리 지표 (distance = 2 → 유클리드 거리 사용)
- ✓ 여섯 번째 인자: 결과 취합 방식 (kernel = "rectangular" → 모든 이웃에 동등한 가중치 부여)



# R 실습: k-NN Classification

## ❖ k-NN 분류 수행

```
# View the k-nn results
summary(kknn)
kknn$CL
kknn$W
kknn$D

table(tstTargets, kknn$fitted.values)
```

- `summary(kknn)`: 검증 데이터에 대한 k-NN 결과물 요약 (예측된 범주, 첫 번째 범주에 속할 확률, 두 번째 범주에 속할 확률, 범주 순서는 알파벳 순)
- `kknn$CL(W/D)`: `kknn`에 저장된 검증 데이터의 Class label(각 이웃의 weight, 각 이웃과의 distance)
- `tables(A,B)`: A와 B의 교차 빈도표 생성 → Confusion matrix 생성

```
> table(valTargets, kknn$fitted.values)
```

valTargets	B	M
B	64	3
M	1	103

# R 실습: k-NN Classification

## ❖ Leave-one-out (LOO) validation을 수행하여 최적의 k값 찾기

```
# Leave-one-out validation for finding the best k
knnttr <- train.kknn(Target ~ ., trnData, kmax=10, distance=2, kernel="rectangular")

knnttr$MISCLASS
knnttr$best.parameters
```

### ■ train.kknn(): LOO validation을 수행하는 함수

✓ 첫 번째 인자: Formula, 두 번째 인자: 학습 데이터

✓ 세 번째 인자: 1부터 1씩 증가시켜가며 탐색할 최대 k값 (이 예시에서는 1, 2, 3, ..., 10, 총 10개의 k값을 탐색)

### ■ \$MISCLASS: 오분류율 확인

### ■ \$best.parameters: 최적 파라미터 조합 결정

```
> knnttr$best.parameters
```

```
$kernel
```

```
[1] "rectangular"
```

```
$k
```

```
[1] 3
```

```
> knnttr$MISCLASS
```

```
rectangular
```

```
1 0.07537688
```

```
2 0.07537688
```

```
3 0.03768844
```

```
4 0.05025126
```

```
5 0.04271357
```

```
6 0.05778894
```

```
7 0.04271357
```

```
8 0.04020101
```

```
9 0.04522613
```

```
10 0.04522613
```

# R 실습: k-NN Classification

## ❖ 최적의 k값을 이용하여 분류 수행

```
# Perform k-nn classification with the best k,
# Distance = Euclidean, and weighted scheme = majority voting
kknns_opt <- kknn(Target ~ ., trnData, tstData,
                  k=knnsr$best.parameters$k, distance=2, kernel = "rectangular")
fit_opt <- fitted(kknns_opt)
cfmatrix <- table(tstTargets, fit_opt)
cfmatrix
```

- fitted(kknns\_opt): kknns\_opt에서 추정된 검증데이터에 대한 예측 범주를 의미

```
> cfmatrix
      fit_opt
tstTargets  B    M
      B   64    3
      M    0  104
```

# R 실습: k-NN Classification

## ❖ 최적의 k값을 이용하여 분류 수행



```
# Summarize the classification performances
Cperf = matrix(0,1,3)
# Simple Accuracy
Cperf[1,1] <- (cfmatrix[1,1]+cfmatrix[2,2])/sum(cfmatrix)
# Balanced correction rate (BCR)
Cperf[1,2] <- sqrt((cfmatrix[1,1]/(cfmatrix[1,1]+cfmatrix[1,2]))*
                  (cfmatrix[2,2]/(cfmatrix[2,1]+cfmatrix[2,2])))
# F1-measure
Recall <- cfmatrix[2,2]/(cfmatrix[2,1]+cfmatrix[2,2])
Precision <- cfmatrix[1,1]/(cfmatrix[1,1]+cfmatrix[1,2])
Cperf[1,3] <- 2*Recall*Precision/(Recall+Precision)
colnames(Cperf) <- c("ACC", "BCR", "F-1")
Cperf
```

- 생성한 confusion matrix를 이용하여 단순 정확도, 균형정확도, F1 지표 계산

```
> Cperf
      ACC      BCR      F-1
[1,] 0.9824561 0.9773556 0.9770992
```

# R 실습: k-NN Regression

## ❖ 데이터: Concrete Strength

	 <a href="#">Concrete Compressive Strength</a>	Multivariate	Regression	Real	1030	9	2007
---	---	--------------	------------	------	------	---	------

- 콘크리트 혼합에 사용되는 재료들의 구성 비중에 따른 콘크리트 강도를 예측

Name -- Data Type -- Measurement -- Description

Cement (component 1) -- quantitative -- kg in a m3 mixture -- Input Variable  
 Blast Furnace Slag (component 2) -- quantitative -- kg in a m3 mixture -- Input Variable  
 Fly Ash (component 3) -- quantitative -- kg in a m3 mixture -- Input Variable  
 Water (component 4) -- quantitative -- kg in a m3 mixture -- Input Variable  
 Superplasticizer (component 5) -- quantitative -- kg in a m3 mixture -- Input Variable  
 Coarse Aggregate (component 6) -- quantitative -- kg in a m3 mixture -- Input Variable  
 Fine Aggregate (component 7) -- quantitative -- kg in a m3 mixture -- Input Variable  
 Age -- quantitative -- Day (1~365) -- Input Variable  
 Concrete compressive strength -- quantitative -- MPa -- Output Variable

# R 실습: k-NN Regression

## ❖ 필요 패키지 설치, 데이터 불러오기 및 전처리

```
# k-Nearest Neighbor Learning (Regression) -----
install.packages("FNN", dependencies = TRUE)
library(FNN)
# Concrete strength data
concrete <- read.csv("concrete.csv", header = FALSE)
RegX <- concrete[,1:8]
RegY <- concrete[,9]
# Data Normalization
RegX <- scale(RegX, center = TRUE, scale = TRUE)
# Combine X and Y
RegData <- as.data.frame(cbind(RegX, RegY))
# Split the data into the training/test sets
set.seed(12345)
trn_idx <- sample(1:1030, round(0.7*1030))
trn_data <- RegData[trn_idx,]
tst_data <- RegData[-trn_idx,]
```

- “FNN”: k-NN regression 알고리즘을 제공하는 패키지
- 학습 및 테스트 용으로 70%와 30%를 무작위로 할당

# R 실습: k-NN Regression

## ❖ Leave-one-out validation 수행

```
for (i in 1:length(nk)){
  cat("k-NN regression with k:", nk[i], "\n")
  tmp_residual <- matrix(0, trn.n, 1)
  for (j in 1:trn.n){
    # Data separation for leave-one-out validation
    tmptrnX <- trn_data[-j, 1:(trn.v-1)]
    tmptrnY <- trn_data[-j, trn.v]
    tmpvalX <- trn_data[j, 1:(trn.v-1)]
    tmpvalY <- trn_data[j, trn.v]
    # Train k-NN & evaluate
    tmp.knn.reg <- knn.reg(tmptrnX, test = tmpvalX, tmptrnY, k=nk[i])
    tmp_residual[j, 1] <- tmpvalY - tmp.knn.reg$pred
  }
  val.rmse[i, 1] <- sqrt(mean(tmp_residual^2))
}
```

### ■ 두 개의 반복문 사용

- ✓ (1) 1부터 k 후보 모두에 대해서
- ✓ (2) 주어진 k값에 대해 모든 학습 데이터에 대해서

# R 실습: k-NN Regression

## ❖ Leave-one-out validation 수행

```
for (i in 1:length(nk)){
  cat("k-NN regression with k:", nk[i], "\n")
  tmp_residual <- matrix(0, trn.n, 1)
  for (j in 1:trn.n){
    # Data separation for leave-one-out validation
    tmptrnX <- trn_data[-j, 1:(trn.v-1)]
    tmptrnY <- trn_data[-j, trn.v]
    tmpvalX <- trn_data[j, 1:(trn.v-1)]
    tmpvalY <- trn_data[j, trn.v]
    # Train k-NN & evaluate
    tmp.knn.reg <- knn.reg(tmptrnX, test = tmpvalX, tmptrnY, k=nk[i])
    tmp_residual[j, 1] <- tmpvalY - tmp.knn.reg$pred
  }
  val.rmse[i, 1] <- sqrt(mean(tmp_residual^2))
}
```

### ■ Leave-one-out

✓ j번째 개체만 검증용으로 설정하고 나머지는 학습용으로 설정



# R 실습: k-NN Regression

## ❖ Leave-one-out validation 수행

```
for (i in 1:length(nk)){
  cat("k-NN regression with k:", nk[i], "\n")
  tmp_residual <- matrix(0, trn.n, 1)
  for (j in 1:trn.n){
    # Data separation for leave-one-out validation
    tmptrnX <- trn_data[-j, 1:(trn.v-1)]
    tmptrnY <- trn_data[-j, trn.v]
    tmpvalX <- trn_data[j, 1:(trn.v-1)]
    tmpvalY <- trn_data[j, trn.v]
    # Train k-NN & evaluate
    tmp.knn.reg <- knn.reg(tmptrnX, test = tmpvalX, tmptrnY, k=nk[i])
    tmp_residual[j, 1] <- tmpvalY - tmp.knn.reg$pred
  }
  val.rmse[i, 1] <- sqrt(mean(tmp_residual^2))
}
```

### ■ knn.reg( ): k-NN regression 수행

- ✓ 첫 번째 인수: 학습 데이터의 설명변수, 두 번째 인수: 테스트 데이터의 설명변수
- ✓ 세 번째 인수: 학습 데이터의 종속변수, 네 번째 인수: k

# R 실습: k-NN Regression

## ❖ Leave-one-out validation 결과 확인

```
val.rmse

# find the best k
best.k <- nk[which.min(val.rmse)]
best.k
```

### ■ 최적 k: 3

```
> val.rmse
      [,1]
[1,] 9.578881
[2,] 9.405462
[3,] 9.377102
[4,] 9.380336
[5,] 9.385702
[6,] 9.488482
[7,] 9.538699
[8,] 9.671501
[9,] 9.706127
[10,] 9.767597
```

```
> # find the best k
> best.k <- nk[which.min(val.rmse)]
> best.k
[1] 3
```

# R 실습: k-NN Regression

## ❖ 최적 k값을 이용하여 모델 학습

```
# Evaluate the k-NN with the test data
test.knn.reg <- knn.reg(trn_data[,1:ncol(trn_data)-1],
                       test = tst_data[,1:ncol(tst_data)-1],
                       trn_data[,ncol(trn_data)], k=best.k)

tgt.y <- tst_data[,ncol(trn_data)]
knn.haty <- test.knn.reg$pred
```

### ■ knn.reg( ): k-NN regression 수행

- ✓ 첫 번째 인수: 학습 데이터의 설명변수
- ✓ 두 번째 인수: 테스트 데이터의 설명변수
- ✓ 세 번째 인수: 학습 데이터의 종속변수
- ✓ 네 번째 인수: k

# R 실습: k-NN Regression

## ❖ 다중선형회귀분석 모형과의 비교

```
# Train the MLR for comparison
full_model <- lm(RegY ~ ., data = trn_data)
mlr.haty <- predict(full_model, newdata = tst_data)

# Regression performance comparison in terms of MAE
mean(abs(tgt.y-knn.haty))
mean(abs(tgt.y-mlr.haty))
```

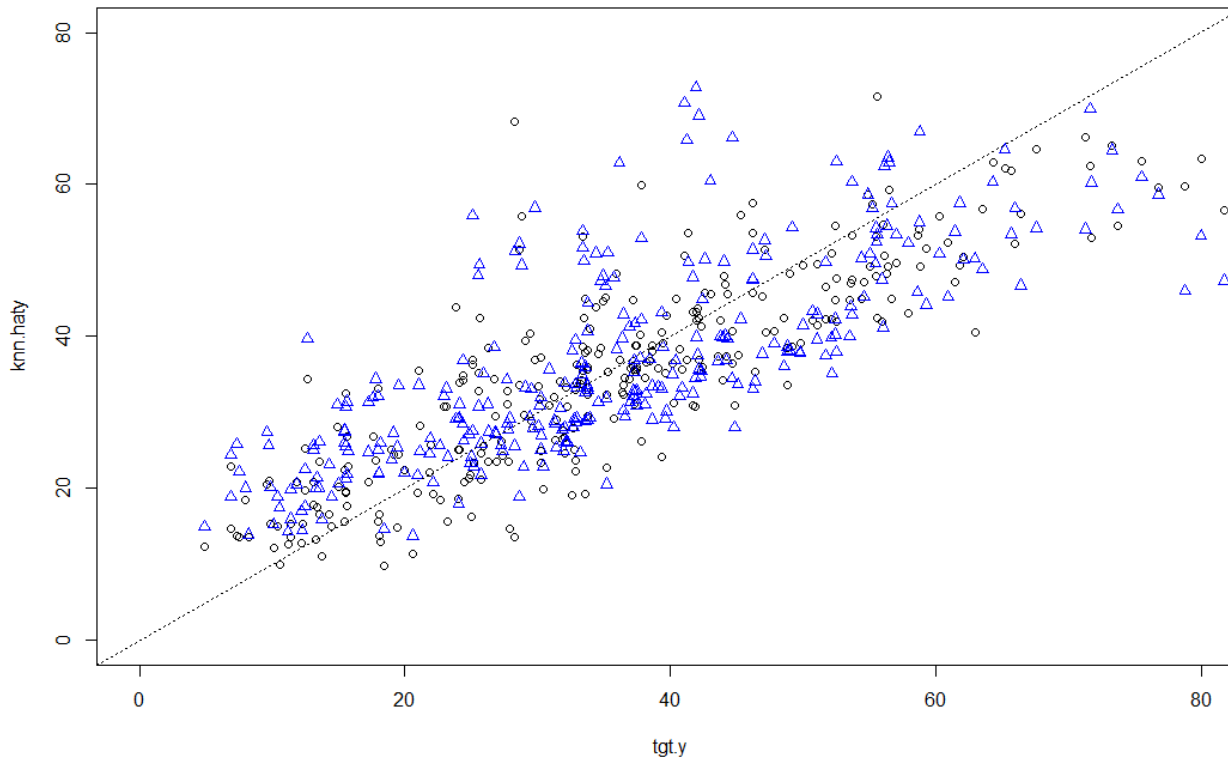
- knn이 MLR보다 낮은 MAE를 나타냄

```
> mean(abs(tgt.y-knn.haty))
[1] 6.462481
> mean(abs(tgt.y-mlr.haty))
[1] 8.166065
```

# R 실습: k-NN Regression

## ❖ 다중선형회귀분석 모형과의 비교

```
# Plot the result
plot(tgt.y, knn.haty, pch = 1, col = 1, xlim = c(0,80), ylim = c(0, 80))
points(tgt.y, mlr.haty, pch = 2, col = 4, xlim = c(0,80), ylim = c(0,80))
abline(0,1,lty=3)
```



△: MLR  
○: k-NN

# Q & A

